International Journal of Research in Computer Science
eISSN 2249-8265 Volume 2 Issue 6 (2012) pp. 21-31
www.ijorcs.org, A Unit of White Globe Publications
doi: 10.7815/ijorcs. 26.2012.052

CAN "FEATURE" BE USED TO MODEL THE CHANGING ACCESS CONTROL POLICIES?

K.Shantha Kumari[1], Dr T.Chithralekha[2]
[1]*Research Scholar, Dept of Banking Technology, Pondicherry University, INDIA*
Email: shanthajayakumar@gmail.com
[2]*Associate Professor, Dept of Computer Science, Pondicherry University, INDIA*
Email: tchitu@yahoo.com

Abstract: *Access control policies [ACPs] regulate the access to data and resources in information systems. These ACPs are framed from the functional requirements and the Organizational security & privacy policies. It was found to be beneficial, when the ACPs are included in the early phases of the software development leading to secure development of information systems. Many approaches are available for including the ACPs in requirements and design phase. They relied on UML artifacts, Aspects and also Feature for this purpose. But the earlier modeling approaches are limited in expressing the evolving ACPs due to organizational policy changes and business process modifications. In this paper, we analyze, whether "Feature"- defined as an increment in program functionality can be used as a modeling entity to represent the Evolving Access control requirements. We discuss the two prominent approaches that use Feature in modeling ACPs. Also we have a comparative analysis to find the suitability of Features in the context of changing ACPs. We conclude with our findings and provide directions for further research.*

Keywords: *Access control polices, Features Functional requirements, modeling, RBAC*

I. INTRODUCTION

Access control is defined as the ability to permit or deny access to a particular resource (object) by a particular entity (subject). An Access control policy defines the (high-level) rules according to which access control must be regulated. An ACP may express conditions that must be satisfied before an access request can be granted. ACPs are derived from requirements as well as high-level security and privacy policies of the organization. Traditionally, ACPs are handled in an ad-hoc way i.e. addressed in an existing system either as an afterthought, or manually injected into practices of the organization. Late analysis of ACPs can generate conflicts between them and the FRs of the system. This leads to security failures, violations from the access control rules, leakage of vital information etc [1]. To properly address security risks and vulnerabilities without jeopardizing speed or cost, organizations must bring security into the development process and this proved to be effective. Hence, the process of integrating the ACPs with the FRs was recommended [2], [3], [4], [5], and [6]].

From the design perspective, access control policies provide an insight to the various kinds of threats, violations that can be handled in the design phase of the software development. The overall system development process is fruitful when the design phase supports integrated modeling of ACPs and FRs. Hence defining the ACPs in the same way as the FRs in the Design phase is considered as a prudent way. The modeling process should be expressive and flexible enough to accommodate all the different requirements that may need to be expressed, while at the same time be simple both in terms of use and implementation (so that it can be verified with ease).

In some prior works, the existing modeling methodologies are modified to define ACPs as like FRs. These methodologies have to take care of the proper abstraction of ACPs and the process of modeling without losing the consistency of the functional requirements. Also, the ACPs of today's enterprise information systems tend to be complex, dynamic and scalable with respect to the resources under the protection, top management's policy changes etc. The Modeling approach has to accommodate the same for providing consistent access control throughout the system development. But the present modeling methodologies cannot support these kinds of intricate requirements. There is a need for higher level of abstraction to capture the complex characteristics of ACPs.

In recent years, Feature oriented programming and Feature modeling caught the attention of the research world. Features- a basic building block that represents the intention of concept can satisfy intuitive user formulated requirements on the software systems. Feature modeling also has the capacity to represent variations by feature composition. This makes it more suitable for modeling dynamic requirement.

In this paper, Feature based ACPs modeling approaches are studied and analyzed. A detailed comparison of each feature modeling approach is done. The advantages and the limitations of every method are presented. This analysis would help for possibilities of new areas in further research. In section 2, the modeling approaches prior to feature modeling are presented and their limitations are reviewed. In Section 3, Features and Feature modeling are introduced. Then ACP modeling approaches using features are analyzed and a comparative study is done in Section 4. The paper concludes with the future directions of research.

II. LITERATURE REVIEW

Olden year software systems, by their very nature, have simple and generic access operations and are not concerned with the meaning of the resources they handle. On the other hand, modern Information sensitive systems and applications handle quite complex access operations due to their specific user purposes and high functionality. This complexity needs to be translated by generic models and specific ACPs into design phase understandable processes. Many approaches including the Formal approaches, High level languages, Model based approaches [UML], Process based approaches are available.

1. Formal logic approaches [[7], [8], [9], [10], [11], [12], [13], [14]] have been a significant part in this research area. They proved to be very much useful due to their formal theory. Approaches for specifying and analyzing the ACPs are based on sophisticated mathematical concepts, that formally stated allows one to check whether the developed ACPs enforce the required level of protection. However, in practice, applying mathematically-based formal specification techniques can be difficult because of the high degree of mathematical skill needed. Thus formal-logic based approaches which are difficult to use and understand, are seldom used by application developers.
2. Another extreme end is High level languages [[15], [16], [17]]. High-level languages are easy to use and understand, but are not amenable for analysis. Therefore, a representation that can be analyzed without sacrificing understandability and usability is desirable.
3. The UML diagrams [3] [18] [19]and UML profiles [4] [5] model the static ACPs. The constraints in the requirements are captured using OCL. The effectiveness of the UML as a standard is predicated. Among other things, being a clear, precise, and pragmatic semantics for its notations is an advantage. But the semi-formal semantics of UML has ambiguity and inconsistency issues. Moreover UML captures the static requirements and represents their behavior using anyone of the diagram. But the scalable, dynamic and evolving natures of the ACPs are not represented in UML. Moreover they either focus on specifying security requirements in UML notation or analyzing UML models against the specified requirements. Still the problem of systematic enforcement of the specified requirements exists. To overcome this, researchers took the idea from “Principle of separation of concerns”. This principle helps to identify, encapsulate and manipulate those parts of software that are relevant to a particular “concern” that may crosscut many design elements at the Design level.
4. Aspect oriented Software development technologies are a promising proposal to enable the modularization of these crosscutting concerns [20]. ACPs being a crosscutting concern across the functional design and implementation can be encapsulated as “aspects”. Subsequently, a weaving process is employed to compose core functionality model elements with those aspects, thereby generating an architecture design. The dynamic nature of ACPs is well addressed by “Aspect oriented modeling” [AOM]. The use of aspects in security is among the most successful uses of aspect-oriented concepts, both at the specification and coding levels. The Aspect oriented software development which uses the aspects as its base unit typically has a programming language character in order to attract a wide user community.
   – Despite its attractiveness to programmers, this code-level approach has its disadvantages. Generally aspects are stated operationally, that results in unclear specification of it intention. It is quite difficult to prove the correctness of an aspect. Also, it is quite unsure about the application of pointcut to all occurrences of the intended runtime events. Moreover the complexities of the system and ACPs may require extending the pointcut and advice description to cover any extra cases which requires an understanding of the modified set of runtime events being targeted, and what code to execute in each specific context of occurrence [21].
5. But a feature represents the intention of a concept. A feature is a logically cohesive piece of functionality and is present in all phases of software development. Features may occur at any level, for example high-level system requirements, architectural level, subsystem and component level. Thus, it is natural to expect that modularization of software into features can provide a lot advantages. ACPs which tend to be continual can be modeled using feature modeling as the features reflect user requirements and can incrementally refine one another.

The available approaches available for access control modeling in the design phase could be classified into the following way as shown in figure 1.

However, in this paper, the emphasis is on delving into the details of Feature Oriented modeling approaches only. This is because, as mentioned earlier, features based approaches are more appropriate for modeling complex and dynamic access control requirements modeling. Hence, the disadvantages of all the other approaches as a whole are explained and the subsequent discussion progresses with the feature oriented modeling.

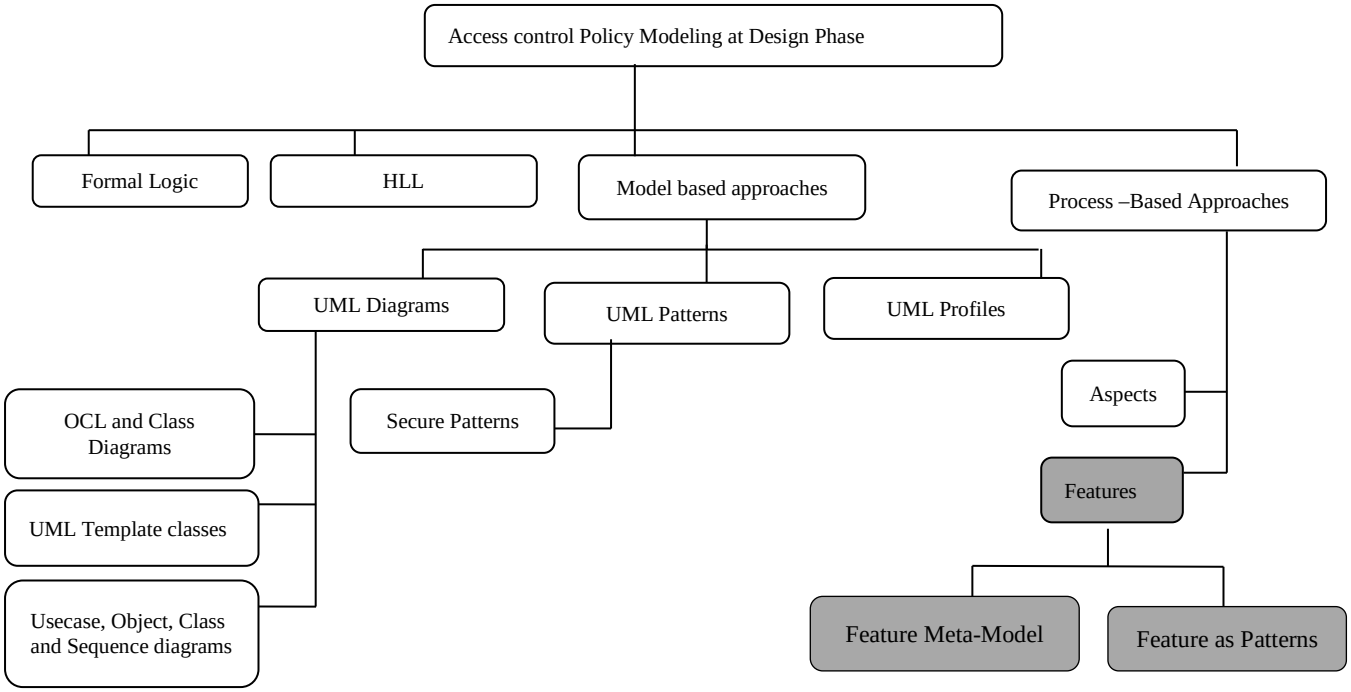

Figure 1: Classification of Different Approaches in ACP Modeling at Design Phase

In the forthcoming sections, a preliminary introduction to Feature, Feature modeling is given. Following is the brief explanation of TWO different approaches taken for modeling the ACPs using Feature. From the Literature survey, it is noted that the Feature based modeling for ACPs is a new emerging area of research and hence limited number of publications are available. The Available publications are either based on designing a Feature meta-model or treating a Feature as a Pattern similar to UML. As the years progress, many research works based on Feature would be available.

III. FEATURES, FEATURE MODELING AND ACPS

Features are basic building blocks that satisfy intuitive user formulated requirements on the software system. They reflect user requirements and incrementally refine one another. Feature model was introduced from the Feature-Oriented Domain Analysis (FODA) methodology [22] and further developed from a number of approaches.

Feature models are used to represent the variability and commonality of software product lines, and permit the configuration of specific applications. Hence it can be defined as the activity of modeling the common and variable properties of concepts [23]. For planning future requirements, the integration of domain analysis activities with software development turned out to be necessary, both from a process and from the economic point of view. Ever since its introduction in 1990, feature modeling has attracted a great number of application domains. And it becomes the most popular method of domain analysis with the development of domain engineering and product line.

Secure Software development is increasingly under the demands of unpredictable, diverse and growing business requirements, as well as limited time-to-market, meticulous product value and mounting competition. The ACPs fall under this category can be elegantly modeled with the FRs using feature modeling

- Feature Meta-modeling is a standard way of representing the ACPs as features and also provides a conceptual scheme to integrate the semiformal requirements with the functional requirements.
- In the same lines, Patterns are also proved to be a wonderful tool for defining the ACPs.
- Techniques like Ontological models and Case-base reasoning is also available as higher abstract structures.

In this paper, Features being an intention of concept is modeled and discussed as Meta-models and Patterns. As mentioned earlier, being a novel thought, Feature based ACP modeling is carried out with these two approaches only till now. The Following section reviews the related work in both the approaches.

A. Meta-Model based ACPs Modeling Approaches

Meta-modeling in software engineering and systems engineering among other disciplines, is the analysis, construction and development of the frames, rules, constraints, models and theories applicable and useful for modeling a predefined class of problems. A meta-model is yet another abstraction, highlighting properties of the model itself. A model conforms to its meta-model.

- Using Meta-modeling concept, the ACPs can be presented as a Feature along with their relationships, dependencies and constraints in a simple yet powerful way. The complexity and the dynamic nature of the ACPs can be captured and represented using composition and decomposition relationships.

- Any Access control requirement represented in the model that conforms to the meta-model ensures the consistency between the FR and ACPs. For example, RBAC model can be used to represent the ACPs with respect to the meta-model. The Core RBAC and other forms of RBAC can be step-wise refined. The RBAC requirements can be composed with the domain requirements as per the composition relations described in the meta-model.

Research work available: In this section, one of the premier works done in modeling the access control requirements using Feature Meta-model is reviewed. The work in [24] presents a feature-based Access Control Requirements Modeling approach. The Access control policy is unfolded as Access control requirement in this approach. Access control requirement and FRs are represented as AC-Feature and F-Feature respectively. Any Access control model including RBAC, DAC, and MAC is abstracted as ACM-Feature. Any Access control requirement is modeled as a composition of ACM and F-Features. The Composition relations are defined in the Feature meta-model. This helps to capture the model to which the ACPs conform and also ensures the consistency between FR and ACPs. This work adapts the Feature meta-model defined in [25]. The easy-to-tailor nature of the feature model can be used to elegantly control the complexity and changefulness of ACPs. The lacking factor of this approach is that, the relationships modeled are very minimum, when compared to the real time. Interactions between features are not considered in this approach.

B. Pattern Based ACP Modeling approaches

A design pattern describes a generic solution to a recurring design problem. ACPs for any domain can be captivated in form for Access control patterns and can be applied along with the software functional requirements. Whereas Meta-Model represents a high level of abstraction of various models, the Patterns can be defined as the solution to a repeated problem with respect to context. Just as Meta-model is tailored to domain specific requirements; patterns too can be instantiated for a particular application. Features as Patterns and its structure can be represented using Feature Modeling. Any Access control model's properties and characteristics can be defined using a declarative language such as OCL, when the Pattern is represented using UML.

Research work available: The works defined in [[26], [27], [28]] have done modeling of Access control policies as patterns and configured with respective domain requirements. ACPs are conformed to RBAC model and then they are treated as a Pattern with OCL defined constrains. The works are briefly explained in the following section:

1. Any Policy is useful only, when it is appropriate at that particular instant. The work in [26] used this concept and presents a modeling approach that configures ACP features on "Need Basis". This method produces Configured ACP features with respect to the Application. ACPs are represented as RBAC features and they are composed with the FRs using "Partial Inheritance". The RBAC features and the relationships are captured by Feature modeling and UML is used for specifying the RBAC features in a form of patterns. The partial inheritance of RBAC features enables step-wise composition, which allows verification of immediate impact of selected features. The authors use the Object Constraint Language (OCL) [29] to define operation semantics. This approach enables fine-grained configuration of RBAC at the feature level in a systematic manner, which helps to lower development complexity and reduce potential errors by excluding unnecessary features. . This work didn't represent all the properties of the Access control model [RBAC] taken for representing the ACPs.

2. The lacking factor of the previous work is resolved in [27]. The authors have extended to include the specification for the static separation of duty feature and also updated the Core & General features with additional behaviors of RBAC. The complete

profile of RBAC is represented as a Pattern. In this pattern Core and non-core features are defined to represent the basic and additional behaviour of the Access control model. Along with Partial inheritance, "Compatibility principle of design" is used to check the Compatible features. This approach benefits most in-house development as it allows feature configuration in full scale to address the specific needs of the target system and seamless integration of configured RBAC into architecture. Still the privacy and temporal features have to be explored and also the systematic integration of dynamic ACPs has to be addressed.

3. All Access control models can be treated as Patterns and the ACPs can be represented using those patterns. Hybrid Access control models were proved to be effective in many instances. Hence they can also be represented as patterns to define ACPs of same kind. This thought was materialized in [28]. RBAC and MAC were considered in this work. Similar to [27], the features are defined and configured with the application design on need basis. "Homogeneous" and "Heterogeneous Composition" are defined as Composition rules for RBAC and MAC features. This approach enables fine-grained configuration of hybrid models of RBAC and MAC at the feature level, which helps reducing development complexity and errors by excluding unnecessary features in early development phase.

IV. DISCUSSION

From the early days, feature modeling is used in various application domains. It has become one of the success methods of domain analysis with the development of domain engineering and product line. Also, enormous tool support is available.

In spite of their success, feature modeling has not seen widespread adoption as a routine part of systems development practice. Previous surveys and our recent review indicate that there have been successes in the application of feature models to problems of secure software systems; yet have certain limitations too, which have to be researched further. This section provides valuable insights of every approach explained earlier, through summarized overview and Qualitative analysis. Also this paper tries to bring out the work still to be explored in ACP Modeling of feature modeling.

In table 1, we revisit concerns raised in previous surveys and identify progress, trends, and remaining challenges in the light of more recent projects through summary analysis. The summary includes the design of the features used in the method, the composition techniques, the verification process, the representation of ACPs in the proposed models and finally the representation of relationships and dependencies. This summary would highlight the application of Features in representing the characteristics and properties of the Access control model taken for representing the ACPs of an application.

Table 1: Summary of Contributions

Approaches / Parameters	Meta-Model based ACPs modeling Ref: [24]	Pattern Based ACPs modeling		
		Ref: [26]	Ref: [27]	Ref:[28]
How a Feature is defined	Feature Meta-Model	Patterns		
Access control model used for ACP representation.	RBAC	RBAC	RBAC	RBAC & MAC
Definition of ACP	ACP is defined as Access control feature	The Access control Model –RBAC is defined as a Pattern.		
Features defined for representing the ACPs	AC-Features F-Features ACM-Features	Core Feature Hierarchy feature General feature Advanced Feature SOD Feature	Core Feature Hierarchy feature General feature Advanced Feature SOD Feature **Review feature** **Temporal feature**	Core Feature Hierarchy feature General feature Advanced Feature SOD Feature
Refinement Relationships defined	Decomposition, Specialization, Characterization	Refinement using Partial inheritance [No Specialization]		

Representation of Refinement relationships	"Relationships" from UML core package	Class diagrams	
Dependencies defined	Requires, excludes, Complex constraints, Protection	Cardinality, Containment	
Representation of Dependencies	"Relationships" from UML core package	Sequence diagrams	
Interpretation of relations, dependencies, constraints	Using the constraints	Object Constraint Language Pre and Post-Conditions	
Feature interaction and behavior	Behaviour and Interaction are defined using the constraints	Not discussed	Compatibility relation is a set of (1) or relations (e.g., (SSD, DSD)) under the same branch, (2) relations of features across branches that can be combined (e.g., (General, DSD)), and (3) implication dependencies.
Composition techniques	Composition is done through integration AC features with F-Features using relationships defined	Step-wise Composition using Partial inheritance. 1. Relationship composition 2. Class Composition 3. Operation Composition	Homogeneous Composition Heterogeneous Composition
Representation of Composition process	Feature Meta-Model	Feature Composition is defined in terms of class diagrams and sequence diagrams. Also, Feature composition is the refinement in view of Multiple inheritances	
Verification process	Reference process is described as a series of steps.	Step-wise Composition and Configuration is the verification process	
Tool Support	Not Available	IBM Rational Software architect [RSA] and Eclipse plugin	
UML Support	UML Core Meta-Model	UML Class Diagrams and Sequence diagrams	
Meta-Model Usage	Yes	NO	
Reusable Features	Yes	Patterns can be configured based on the application	
Notation of Feature Model	Tree structure	Class Diagrams	
Role Hierarchies	Easily customizable	The features have to configured depending upon the needs	
Feature Conflicts and Interactions	They are not analyzed	They are not analyzed	

Following is the Qualitative analysis of the approaches for ACP modeling in table 2. An analytical framework is partially adapted from the work [30] and [31] , that defines the following technical criteria is used for analysis. Certain criteria are newly proposed in this paper that would enhance the qualitative analysis.

A. Factors that focus on representation of ACPs-Applicability

1. *Expressiveness [31]:* Refers to the ability of representing security policies as models. This is an important criterion that would make the modeling approach successful. Regarding this criterion, any specification approach will evaluated in expressing

the kind of requirements and can take one of the following values:

- Static ACPs: if it allows the specification of the majority of statically enforceable ACPs.
- Dynamic ACPs: if it allows the specification of the majority of dynamically enforceable ACPs.
- Evolving ACPs: if it allows the specification of majority of evolving new ACPs.
- Scalable ACPs: if it supports scalability of the ACPs.
- All ACPs: if it supports all ACPs

2. *Level of formality [30]:* Refers to the ability to check the formality [Procedures] in Modeling approaches, to design and produce clear, reliable, complete and provable Models for the expressed ACPs.
   - Clear design: if the procedure produce a unambiguous design
   - Reliable design: if the procedure supports consistent design
   - Complete design: if the approach produces a complete model
   - Provable design: if the approach produces a model that can be verified.
3. *Scope of specifications [30]:* Refers to the ability of representing both ACPs and FRs using same notations. It can be measured with the value "complete".
4. *Comprehensiveness:* Refers to the ability to check whether the particular set of modeling notations and parameters including relationships are comprehensive to represent the ACPs. This can take the following values
   - Complete :if the approach can model all kinds of ACPs and relationships
   - Incomplete: if the approach fails to capture certain ACPs and their relationships.

B. Factors that Focus on Usability of the Approaches:

1. *Learn ability:* Refers to the ability of learning the modeling approach and using it. It can take the following values:
   - Difficult : if the modeling approach is difficult to learn and use for designing
   - Medium: if the approach is learnable and usable after meticulous study and training
   - Simple: if the approach is suitable for usage even for new users.
2. *Usability:* The usability principle measures how simple or complex a technique can be used. In the context of evaluation, the amount of work involved in applying the technique, gathering information and processing for output is considered. Once the information is collected, how usable is the process of designing a diagram, tree, or table. It can take the following values:
   - Difficult: if the modeling approach consists difficult, long procedures/methodology
   - Medium: if the approach consists of intermediate level of methodology with respect to time and work.
   - Simple: if the approach consists of simple methodology
3. *Analyzability:* The analyzability measures how easily a developer can interpret and implement the results provided by the techniques. For example, whether the conflicts between the ACPs and FRs are modeled? Is it possible to verify the security assurance provided by the technique?
   - Easy: Analysis of the Output of the Modeling approach is Simple
   - Difficult: if the output interpretation is difficult.

C. Factors that check the Verifiability of the Models produced by the approaches:

1. *Support for Variations:* Refers to the ability in providing representations to show all possible variations and their output.
2. *Support for Consistency checking:* A Modeling approach should produce a Model that allows checking the consistency between the ACPs and FRs composed together. Automatic **internal** verification supported by a technique may improve the reduction of ambiguity, ensure completeness, improve consistency, and hence, make specifications more verifiable.
3. *Support for Correctness of Specification:* It may ensure correctness of the design produced by the **approach** by validating it against requirements and/or implementation.

Table 2 allows us to relate Modeling approaches and specification technique criteria. We can see, for instance, that the fulfillment of a technical criterion must generate the fulfillment of all Requirement criteria related to that criterion. The degree of fulfillment will be "Y" (Yes), "N" (No) or "P" (Partial).

Table 2: Qualitative Analysis of Features in ACPs modeling

Technique Criterion	***Requirement Criteria***	***Methodologies***	
		Meta-Model based Modeling	***Pattern Based modeling***
Expressiveness	Static ACPs	Y	Y
	Dynamic ACPs	P	N
	Evolvable ACPs	P	N
	Scalable ACPs	N	N
	All ACPs	N	N
Level of Formality	Clear	Y	Y
	Reliable	P	Y
	Complete	P	P
	Provable	P	P
Scope of Specification	Complete	P	P
Comprehensiveness	Complete	P	P
	Incomplete	Y	Y
Learn ability	Difficult	P	P
	Medium	Y	Y
	Easy	N	N
Usability	Difficult	P	P
	Medium	Y	Y
	Easy	N	N
Analyzability	Difficult	P	P
	Easy	P	P
Support for Variations	-	N	N
Support for Consistency checking	-	P	P
Support for Correctness of specification	-	P	P

V. INFERENCE

From Literature survey, Feature modeling using Meta-Model or patterns was found to be optimal and efficient method that supports complex ACPs without much cost and wastage of time. The Summary table highlighted the building blocks of each approach. All the proposed ideas are very interesting and they provide important contributions to solve the security problem in a methodological approach. Nevertheless, they have a series of limitations. The performance of each approach is considered by the way, how the concept of "Feature" is handled in modeling ACPs with FRs.

The Comparative analysis is made with factor concerning to the Usability, Applicability and Verifiability of the approaches. The results are analyzed with respect to the kind of feature and relations defined in the techniques.

A. Feature Behaviour: A feature has the capacity to behave differently in several instances. This was well illustrated in [23]. Feature behaviour or type determines the kind of variability provided at the time of assembling with another feature in an application. Hence the behaviour description is relevant when the feature is ready to compose with another in order to produce a model.

1. In both approaches, the feature is defined a type that signifies how the feature adds up to the variability of the system. This confines the prospects to reuse the same feature in a dissimilar perspective with different variability requirements. For E.g.: An *Amount transfer payment feature* may be mandatory in one context while optional in another [depending upon the target application]. As the type is inextricably associated with the feature, it will not be possible to reuse the feature as it is.
2. In both the approaches, the variant behaviour of the feature is not explored completely. The Feature behaviour is defined statically. This reduces the purpose of reusability and also the support for variation.

B. Feature Relationships: Within a feature model the features are structured by relations. Common to all methods mentioned above are hierarchical relations between a feature and its sub features. They control

the inclusion of features to instances. If an optional feature is selected for an instance, then all mandatory sub-features have to be included as well, and optional sub-features can be included.

1. The "Expressiveness" factor in the qualitative comparison entirely depends on the kind of relationships the approaches support. The Complex ACPs could not be expressed completely with the relationship set defined in both the approaches.
2. The dependencies and the refinement relationships should be expanded to include the complex ACPs.

C. *Feature Conflicts and Interactions*: A feature interaction occurs whenever one feature affects the behaviour of another. The feature interaction problem is generally associated with conflicting features causing undesirable effects. The feature interaction problem is how to rapidly develop and deploy new features without disrupting the functionality of existing features. So this concept has direct relationship with Consistency between the ACPs and FRs. From the summary, it was found that feature interactions were not handled in both the approaches. Hence the support for Consistency check is not completely defined in both techniques.

D. *Feature Decomposition*: In both the approaches, the Features are represented in Top-down approach [Hierarchical]. This poses a problem that the problem domain should be fully understood in prior, to be able to decompose into smaller problems. This needs a combination of Top-down and bottom up approach adaptation that could help in composing and reusing features from an existing and continuously growing storage of features.

E. *Functional and Variability Decomposition*: In the feature models presented, the top down decomposition is implicitly based on Functional and Variability Decomposition. But they are not clearly distinguished, so the modeling process has become very complex with respect to Consistency factor.

F. *Scalability*: Feature meta-model approach have significant characteristics to support Scalability of features, when compared to Pattern based Models. But the tree structure used in Meta-Model based ACP modeling would lose the added value of a graphical representation when trees become very large and easy over viewing is not possible. There is a need for Suitable Abstraction.

VI. CONCLUSION

This paper explores the Feature modeling approaches used for defining and integrating the ACPs with the functional requirements. Also it surveys the related state of the art. A detailed discussion for the explained approaches was presented. This includes the concise review and the Comparative analysis of the approaches. From this paper, it is understood that features are efficient in modeling the ACPs when they tend to be complex. Two approaches based on features were studied and analyzed. They present systematic way for representing ACPs, yet have certain limitations. Based on the limitations, there is a need for complete feature definition including more relationships and constraints that represent the real time security requirements and their complexities. Moreover, the Feature definition should also be able to manage the feature interactions and conflicts to support the consistency. And, to increase the variability options the type of the features has to be defined separately from feature definition. This would support higher level of reusability and variation productions. The Step-wise refinement of features can be strongly explored to adapt for complex and dynamic requirements. The main contribution of this work is the study and discussion of two important Feature modeling methodologies used for the development of the software systems with the proper and systematic integration of ACPs. The major limitation of this

Work is the inadequate availability of research works based on Feature. We reviewed available works in those methodologies and found the certain observations that would propel further research.

VII. REFERENCES

[1] G Georg, I Ray, and R France, "Using aspects to design a secure system," *In Proceedings of the International Conference on Engineering Complex Computing Systems (ICECCS 2002), Greenbelt, MD, ACM Press.*, 2002. doi: 10.1109/ICECCS.2002.1181504

[2] T Doan, S Demurjian, C T Ting, and C Phillips, "RBAC/MAC security for UML," in *Research Directions in Data and Applications Security XVIII ,IFIP International Federation for Information Processing Volume 144*. Catalonia, Spain: Springer, 2004, pp. 189-203.

[3] D Kim, I Ray, R France, and N Li, "Modeling Role-Based Access Control Using Parameterized UML Models ," in *FASE 2004, LNCS, vol. 2984*, 2004, pp. 180–193. doi: 10.1007/978-3-540-24721-0_13

[4] J Jurjens, "UMLsec: Extending UML for Secure Systems Development," in *Proceedings. of the 5th International Conference on the UML*, Dresden, Germany, 2002, pp. 412–425. doi: 10.1007/3-540-45800-X_32

[5] T Lodderstedt, D Basin, and J Doser, "Secureuml: A UML-based modeling language for model-driven security," in *Proceedings of the International Conference on t he Unified Modeling Language, UML'2002*, 2002, pp. 426-441.

[6] T Priebe, E Fernandez, J Mehlau, and G Pernul, "A Pattern System for Access Control," in *Proceedings of Conference on Data and Application Security*, 2004, pp. 22–28. doi: 10.1007/1-4020-8128-6_16

[7] Steve Barker, "Security Policy Specification in Logic," in *Proceedings of the International Conference on Artificial Intelligence,ICAI'2000*, Las Vegas, NV, 2000, pp. 143-148.

[8] Steve Barker and Arnon Rosenthal, "Flexible security policies in SQL," in *Proceedings of the fifteenth annual working conference on D atabase and application security*, Niagara, Ontario, Canada, 2001, pp. 167-180.

[9] Elisa Bertino, Piero Andrea Bonatti, and Elena Ferrari, "TRBAC: a temporal role-based access control model," in *RBAC '00 Proceedings of the fifth ACM workshop on Role-based access control*, Berlin, Germany, 2000, pp. 21–30. doi: 10.1145/501978.501979

[10] Fang Chen and Ravi S. Sandhu, "Constraints for role-based access control," in *RBAC '95 Proceedings of the first ACM Workshop on R ole-based access control* , 1995, p. Article No. 14. doi: 10.1145/270152.270177

[11] R J Hayton, J M Bacon, and K Moody, "Access control in open distributed environment," in *In IEEE Symposium on Security and Privacy* , Oakland, CA, 1998, pp. 3–14. doi: 10.1109/SECPRI.1998.674819

[12] Michael Hitchens and Vijay Varadharajan, "Tower: A Language for Role-Based Access Control," in *POLICY '01 Proceedings of the International Workshop on Policies for Distributed Systems and Networks*, Bristol, U.K. , 2001, pp. 88 - 106. doi: 10.1007/3-540-44569-2_6

[13] S Jajodia, P Samarati, and V S Subrahmanian, "A Logical Language for Expressing Authorizations," in *IEEE Symposium on Security and P rivacy, pages* , Oakland, CA, 1997, pp. 31–42. doi: 10.1109/SECPRI.1997.601312

[14] R Ortalo, "A Flexible Method for Information Systems Security Policy Specification," in *Proceedings of the 5th European Symposium on Research in Computer Security*, Louvain-la-Neuve, Belgium, 1998. doi: 10.1007/BFb0055856

[15] J A Hoagland, R Pandey, and K N Levitt, "Security Policy Specification Using a Graphical Approach," Computer Science Department, University of California, Davis., Technical Report 1998.

[16] OASIS. (2002). Available: http://www.oasis-open.org/committees/xacml.

[17] C Ribeiro, A Zuquete, and P Ferreira, "SPL: An Access Control Language for Security Policies with Complex Constraints," in *roceedings of the Network and Distributed System Security Symposium*, San Diego, CA, 2001.

[18] I Ray, N Li, R. B France, and D. K Kim, "Using UML to visualize role-based access control constraints," in *Proceedings of the Symposium on A ccess Control Models and Technologies(SACMAT)*, 2004, pp. 31-40. doi: 10.1145/990036.990054

[19] I Ray, N Li, D. K Kim, and R. B France, "Using parameterized UML to specify and compose access control models," in *Proceedings of the 6th IFIP TC-11 WG 11.5 Working Conference on Integrity and Internal Control in Information Systems, IICIS'03,* Lausanne, Switzerland, (2003)., 2003. doi: 10.1007/1-4020-7901-X_4

[20] R Filman, T Elrad, S Clarke, and M. Aksit, *Aspect-Oriented Software Development.*: Addison Wesley.,ISBN-10: 0321219767 | ISBN-13: 978-0321219763., 2000.

[21] D R Smith, "A Generative Approach to Aspect-Oriented Programming," in *Proceedings of the Third International Conference on G enerative Programming and Component Engineering (GPCE'04)*, p. 2004. doi: 10.1007/978-3-540-30175-2_3

[22] Kyo C. Kang et al., "FORM : A feature-oriented reuse method,Volume 5," *Annals of Software Engineering* , pp. 143 - 168, 1998.

[23] L Abo Zaid, F Kleinermann, and O De Troyer, "Feature Assembly Framework: Towards Scalable and Reusable Feature Models," in *Proceedings of the 5th Workshop on Variability Modeling of Software-Intensive Systems*, Namur, Belgium, 2011, pp. 1-9.

[24] Lianshan Sun and Gang Huang, "Modeling Access Control Requirements in Feature Model," in *APSEC '09 Proceedings of the 2009 16th Asia-Pacific Software Engineering Conference*, Penang, 2009, pp. 241-248. doi: 10.1109/APSEC.2009.21

[25] Hong Mei, Wei Zhang, and Haiyan Zhao, "A Metamodel for modeling system Features and their refinement, constraint and interaction relationships," *Software and Systems Modeling 5(2)*, pp. 172-186, 2006. doi: 10.1007/s10270-006-0004-1

[26] Dae-Kyoo Kim, Lunjin Lu, and Sangsig Kim, "A Verifiable Modeling Approach to Configurable Role-Based Access Control," in *Proceedings of Fundamental Approaches to Software Engineering (FASE/ETAPS 2010)*, Paphos, Cyprus, 2010, pp. 188-201. doi: 10.1007/978-3-642-12029-9_14

[27] S Kim, D. K Kim, L Lu, S Kim, and S Park, "A Feature-Based Approach for Modeling Role-Based Access Control Systems;," *Journal of Systems and Software Vol. 84, No. 12*, pp. 2035-2052, 2011. doi: 10.1016/j.jss.2011.03.084

[28] S Kim, D.-K Kim, L Lu, S Park, and S Kim, "A Feature-Based Modeling Approach for Building Hybrid Access Control Systems," in *5th International Conference on Se cure Software Integration and Reliability Improvement (SSIRI)*, Jeju, Korea, 2011, pp. 88-97. doi: 10.1109/SSIRI.2011.16

[29] Gail-Joon Ahn and Michael E. Shin, "Role-Based Authorization Constraints Specification Using Object Constraint Language," in *WETICE '01 Proceedings of the 10th IEEE International Workshops on E nabling Technologies: Infrastructure for Collaborative Enterprises*, Washington, DC, USA, 2001, pp. 157 – 162.

[30] A Khwaja and J Urban, "A Synthesis of evaluation criteria for software specifications and specification techniques," *International Journal of Software Engineering and Knowledge Engineering*, vol. 12 , no. 5, pp. 581–599, 2002. doi: 10.1142/S0218194002001062

[31] C Talhi et al., "Usability of Security Specification Approaches for UML Design: A Survey," *Journal of Object Technology*, vol. 8, no. 6, pp. 103-122, 2009. doi: 10.5381/jot.2009.8.6.a1

How to cite

K.Shantha Kumari, Dr T.Chithralekha, "Can "Feature" be used to Model the Changing Access Control Policies?". *International Journal of Research in Computer Science*, 2 (6): pp. 21-31, November 2012. doi:10.7815/ijorcs.26.2012.052